Server-side scripting

Server-side scripting is a technique used in web development which involves employing scripts on a web server which produces a response customized for each user's (client's) request to the website. Scripts can be written in any of a number of server-side scripting languages that are available. Server-side scripting is distinguished from client-side scripting where embedded scripts, such as JavaScript, are run client-side in a web browser, but both techniques are often used together. The alternative to either or both types of scripting is for the web server itself to deliver a static web page.

Server-side scripting is often used to provide a customized interface for the user. These scripts may assemble client characteristics for use in customizing the response based on those characteristics, the user's requirements, access rights, etc. Server-side scripting also enables the website owner to hide the source code that generates the interface, whereas, with client-side scripting, the user has access to all the code received by the client. A downside to the use of server-side scripting is that the client needs to make further requests over the network to the server in order to show new information to the user via the web browser.[1] These requests can slow down the experience for the user, place more load on the server, and prevent the use of the application when the user is disconnected from the server.

When the server serves data in a commonly used manner, for , according to the HTTP or FTP protocols, users may have their choice of a number of client programs (most modern web browsers can request and receive data using both of those protocols). In the case of more specialized applications, programmers may write their own server, client, and communications protocol, that can only be used with one another.

Programs that run on a user's local computer without ever sending or receiving data over a network are not considered clients, and so the operations of such programs would not be considered client-side operations.

History

Netscape introduced an implementation of JavaScript for server-side scripting with Netscape Enterprise Server, first released in December, 1994 (soon after releasing JavaScript for browsers).[2][3]

Server-side scripting was later used in early 1996 by Fred Dufresne while developing the first website for Boston, MA television station WCVB. The technology is described in US patent 5835712 (http://patft.uspto.gov/netacgi/nph-Parser?Sect2=PTO1&Sect2=HITOFF&p=1&u=/netahtml/PTO/search-bool.html&r=1&f=G&l=50&d=PALL&RefSrch=yes&Query=PN/5835712) . The

patent was issued in 1998 and is now owned by Open Invention Network (OIN). In 2010, OIN named Fred DuFresne a "Distinguished Inventor" (https://web.archive.org/web/20111110192158/http://openinventionnetwork.com/press_release06_07_11.php) for his work on server-side scripting.

Explanation

In the earlier days of the web, server-side scripting was almost exclusively performed by using a combination of C programs, Perl scripts, and shell scripts using the Common Gateway Interface (CGI). Those scripts were executed by the operating system, and the results were served back by the web server. Many modern web servers can directly execute on-line scripting languages such as ASP, JSP, Perl, PHP and Ruby either by the web server itself or via extension modules (e.g. mod_perl or mod_php) to the webserver. For example, WebDNA includes its own embedded database system. Either form of scripting (i.e., CGI or direct execution) can be used to build up complex multi-page sites, but direct execution usually results in less overhead because of the lower number of calls to external interpreters.

Dynamic websites sometimes use custom web application servers, such as Glassfish, Plack and Python's "Base HTTP Server" library, although some may not consider this to be server-side scripting. When using dynamic web-based scripting techniques, developers must have a keen understanding of the logical, temporal, and physical separation between the client and the server. For a user's action to trigger the execution of server-side code, for example, a developer working with classic ASP must explicitly cause the user's browser to make a request back to the webserver.

Server-side scripts are completely processed by the servers instead of clients. When clients request a page containing server-side scripts, the application server processes the scripts and returns an HTML page to the client.

Server-side rendering

With the advent of single-page applications, an updated approach to server-side scripting was introduced, known as server-side rendering. With server-side rendering, static HTML can be sent from the server to the client, and client-side JavaScript then makes the web page dynamic by attaching event handlers to the HTML elements in a process called hydration. Examples of frameworks that support server-side rendering are Next.js, Nuxt.js, Angular.

An alternative to server-side rendering is static site generation. With server-side rendering, the page is generally assembled on the server once per each request. By contrast, with static site generation, the application generates all the static HTML pages at build time. These pre-built

pages are then sent to the client at each request, without having to build them again. This leads to faster rendering times, since a new page doesn't have to be generated on every request; the server simply hands over a file it already generated in the past. The downside is that when some data on the page changes, it needs to be rebuilt. Examples of static site generation tools are Jekyll and Gatsby. Sites generated by static site generation tools are often hosted on Netlify, Vercel or GitHub pages.

See also

- Client-side scripting
- Content management system (CMS)
- Edge Side Includes
- JSP
- Node.js
- Next.js
- Outline of web design and web development
- Perl/Plack
- PHP
- Server Side Includes (SSI)
- Web development

References

1. "What do client side and server side mean? Client side vs. server side" (https://www.cloudflare.com/en-gb/learning/serverless/glossary/client-side-vs-server-side/) . Cloudflare. Retrieved 17 April 2025.
2. "Server-Side JavaScript Guide" (http://docs.oracle.com/cd/E19957-01/816-6411-10/getstart.htm) . Netscape Communications Corporation. 1998. Retrieved 2012-04-25.
3. Mike Morgan (1996). "Using Netscape™ LiveWire™, Special Edition" (http://vampire.rulez.org/onlinedoc/book/NetscapeLiveWire/ch6.htm) . Que.

External links